Exhibit I

Telcordia™ Technologies

Formerly Bellcore...
Performance from Experience

Applied Research

Subject: **SLA Monitoring Architecture and Solutions**

Date: **June 14, 1999**

Contact: **Richard Lau**
NVC 3Z319
732-758-5476
clau@telcordia.com

Authors: *Arturo Cisneros*
Krishna Kant
Richard Lau
Christian Rad
Bruce Siegell

Technical Document
IM-583

ABSTRACT:
Virtual Private Networks (VPN) and Customer Network Management (CNM) features are becoming increasingly important in key high-growth service networks based on Wide Area Network (WAN) connectivity. This includes Inter-exchange Carriers (IXC), Telcos (PTT), Local Exchange Carriers (LEC), Internet Service Providers (ISP), and the emerging Integrated Communications providers (ICP) such as Sprint and Level 3 Corp., as well as enterprise networks of large corporations. The support for customizable management partitions is essential to support VPN services. Today's network environment requires services that span multiple network providers and multiple services. In these markets, it is necessary for service providers to provide network management and Service Level Agreements that support partitioned as well as end-to-end views of the network, where partitions may be defined according to geographic, technological, or administrative criteria.

In this document we provide solutions to the SLA measurement problem by examining different schemes and methodologies. The key results are:

- A description of the current direction in IP VPN architecture and its implications on SLA monitoring
- A survey of the industry with respect to SLA monitoring schemes and a gap analysis on what is still missing
- Proposed solutions and their impact on Telcordia SLAM
- New results in SLA monitoring based on passive monitoring; this result is potentially patentable

- Opportunities for Telcordia SLA management OSSs

Table of Contents

List of Figures

List of Tables

List of Equations

1 INTRODUCTION

The nature of packet switched technology has complicated the ability of the IT manager to monitor the performance of their WAN service provider. Hence, performance guarantees have emerged as a means for IT Managers to ensure that their critical business data is delivered in a reliable, consistent manner. These performance guarantees, coupled with traditional support guarantees such as Mean Time to Repair (MTTR) and Mean Time Between Failures (MTBF) are now referred to, in the industry, as Service Level Agreements (SLA). In order to provide SLA capabilities to its IT customers, Telcordia Technologies has embarked on the development of a unique set of features that can support its various customers' network topologies. This demands a careful study of SLA metrics collection methodologies that would provide a comprehensive view and value not only to service providers but also to service receivers, namely, the customers.

1.1 Purpose

This document defines the SLA metrics collection schemes and related methodologies, which supports IP VPN with specific network topologies outlined in Section 3. This document satisfies the second deliverable under the IR&D project (R92504-01) on "SLA for Next Generation Networks"

1.2 Organization

Section 1 (Introduction) provides an overview while Section 2 describes the SLA metrics and their associated definitions. Section 3 and 4 provides SLA metric measurement architecture and solutions to the SLA measurement problem, respectively. Section 5 describes the positioning of Telcordia SLA Management OSS. And Section 6 gives the conclusion. Appendix A provides descriptions of different monitoring technologies. Appendix B outlines an algorithm for framing a stream of IP packets at network ingress and egress points using passive monitors.

1.3 Audience

The primary audience for this document is Telcordia Software Systems.

1.4 Terminology

Terminology is provided throughout the document.

2 SLA METRICS

This section will provide the SLA parameters and associated definitions relevant to IP VPN services. The primary objective is to provide the necessary SLA information for VPN service providers.

2.1 MTTR

The MTTR (Mean Time To Repair) is a SLA parameter that is based on network outage. This parameter must be measured in accordance with the physical layer alarm pickup from the network element in accordance with T1.231 Standard.

2.2 MTBF

The MTBF (Mean Time Between Failures) is a SLA parameter that is based on the time between the starting times of consecutive network outages averaged over a long measured period of time.

2.3 Throughput

IP packet throughput at an egress Measurement Point[1] (MP) is defined as the total number of IP packets (in bits) observed at that egress MP during a specified time interval divided by the time interval duration (equivalently, the number of successful IP packet (bits) transfers per service-second.)

2.4 Utilization

Is defined as throughput divided by the contracted throughput expectation (allocated rate).

2.5 Packet Loss Ratio (PLR)

PLR is defined as the difference between the packet counts at the source and destination MPs divided by the packet count at the source MP for a measured interval of time

[1] The Measurement Point (MP), based on ITU-T Recommendation I.380, is defined as the boundary between a host and an adjacent link at which performance reference events can be observed and measured.

2.6 Packet Delay

Packet delay is defined as the difference of UTC[2] time at the source and destination MP for a given uniquely identifiable packet for transmission. The measurements associated with VPN services have been round-trip delays in today's environment and technological capabilities.

2.7 Availability

Availability is the percentage of time that the IP service is available. The basis for service availability function is a threshold on the IP Packet Loss Ratio (IPLR) performance. The IP service is available on an end-to-end basis if the IPLR for that end-to-end case is smaller than the threshold defined by the customer.

2.8 Packet Jitter

Defined as the difference between delays for a pair of consecutive packets that are observed at source and destination points

3 SLA METRIC MEASUREMENT ARCHITECTURE

This section will provide the SLA metric measurement architecture with respect to the IP VPN service topology.

3.1 Service Types

This section describes two common types of services, which have direct impact on the SLA measurement architecture. The first type, which we call ***Type I***, is an Internet access service in which the end user accesses Internet (customer prem to the ISP network) services across various administrative domains (or jurisdictions) as shown in Figure 1 below. Examples of this service include www and IP telephone. There are multiple access technologies including ADSL, dedicated private lines, or SONET for supporting this service. In addition, end-to-end management becomes complex as multiple technologies such as Frame Relay, ATM, and IP networks form islands of administrative and operations domains. Coordination among administrative domains will be required to manage the end-to-end aspect of this service as described in Section 4.

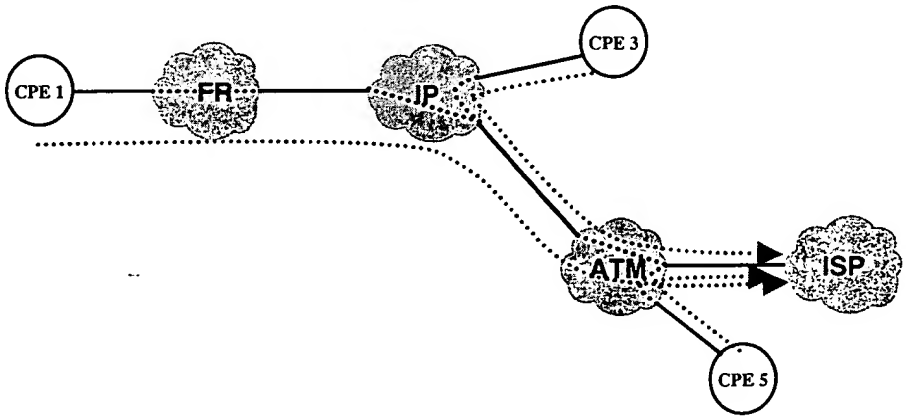

Figure 1: Type I Network Topology for IP Services

As seen depending on the technologies on the customer premises, the service providers must comply with access technology that can provide the given service. However the critical issue with every service provider is the management of the IP network within its domain.

The second type of service, which we call ***Type II***, is a virtual private network service offered within the administrative domains of the service provider (customer_prem-to- customer_prem across a service provider network domain). In this case, the service provider has end-to-end management responsibility. This service can be offered as an ATM cell relay service or as an IP Virtual Private Network. As shown in Figure 2 below, the network is made-up of clusters of technology domains, which make up the paths for the IP VPN services. Although the predominant access technology in the service provider's domain is Frame Relay (FR), customers

[2] UTC: Universal Time Coordinated

may have varied technologies on their premises that cannot be precluded from entering the service provider's domain.

Again as was the case with the Type I topology, coordination among administrative domains will be required to manage the end-to-end aspect of this service.

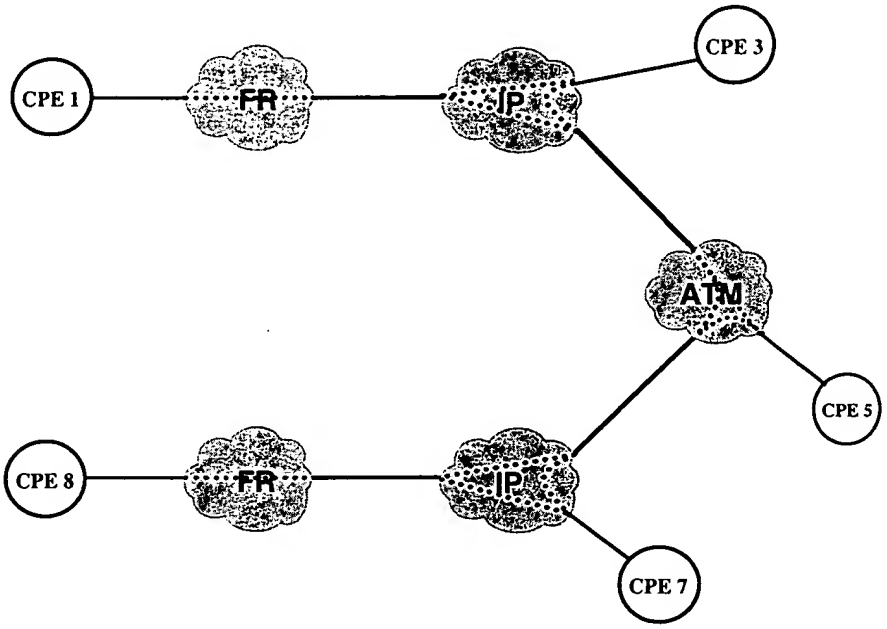

Figure 2: Type II Network Topology for IP VPN Services

3.2 Measurement Point

The Measurement Point (MP), based on ITU-T Recommendation I.380 [I.380], is defined as the boundary between a host and an adjacent link at which performance reference events can be observed and measured. It should also be noted that a section[3] or a combination of sections is measurable if it is bounded by a set of MPs. Also, depending on the type of the service, an MP may involve different observable layers.

4 SOLUTIONS TO THE SLA MEASUREMENT PROBLEM

The primary objective of any service provider is to provide a quality service to their customers. However, the term quality has become a difficult one to define and to implement in light of today's complex network environments. The network environment is multi-vendor and made up of different types of equipment with different types of statistics. True end-to-end statistics are difficult to measure and correlate. In this section we look at different scenarios and methodologies to support IP VPN measurements.

4.1 CPE Based Solution

The CPE (Customer Premises Equipment) based solution is perhaps the most accurate and effective means of obtaining end-to-end SLA parameters. Solutions, such as Surveyor and RTR, already exist for obtaining these measurements (see Appendix A). However, there are several issues that must be considered regarding this methodology:

- SLA negotiation with both the access provider as well as Inter Exchange carriers

[3] A section is defined as the link connecting i) a source or destination host to its adjacent host (e.g. router), possibly in another jurisdiction, or ii) a router in one network section with a router in another network section.

- Management of the customer premises equipment. This is either performed by the customer or by the service provider
  - A network element managed by the customer will require a unique set of SLA negotiation with the service provider(s) that can correlate the customer measurements with those of service providers
  - A customer network element managed by the service provider will require access to the measuring equipment on the customer premises by the service providers Operations System
  - Cost associated with either one of these scenarios can become prohibitive, since in the first case correlation and resolution of discrepancies is a non-trivial negotiating feat. And, in the second case network service providers would not only have to make the measurements from customer_prem-to-customer_prem, but also make measurements within their own network domain.

In any event, a service provider will need to make measurements within its own service domain in order to determine the source of any non-compliance with the SLA parameters.

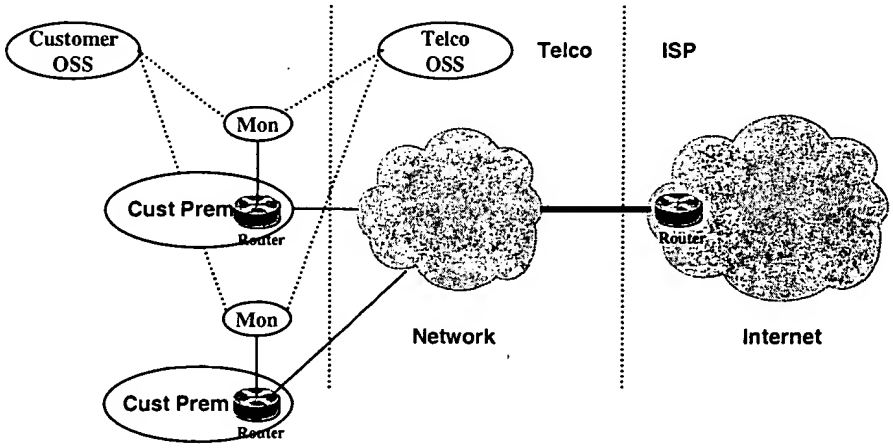

Figure 3: A network topology utilizing end-to-end measurement scheme

4.2 Measurement at network boundary

Many SLA monitoring devices to-date focus on solving the problem of monitoring and collecting statistics with respect to a specific technology or layer. One example is Visual UpTime from Visual Networks [VisualNetworks99], which supports monitoring of SLA parameters for Frame Relay networks (Layer 2) or ATM networks (Layer 2) but offers no IP VPN correlation. Similarly, OCnMON [Apisdorf96] from MCI/WorldCom can be used to observe ATM and IP packets at a single measurement point (passively), but is not capable of correlating IP statistics at different points of a VPN. Another example is Cisco's Response Time Reporter (RTR) [Cisco99], a component of the IOS operating system on Cisco routers, that actively measures SLA metrics at the IP level, but only works at the IP level. In many practical network scenarios, however, measuring SLA parameters at a specific layer is not sufficient. For example, knowing the SLA metrics with respect to a Frame Relay network will not be sufficient for reporting end-to-end SLA metrics for a VPN that connects two CPE locations. In this case, RTR located at the CPE will work, but, as mentioned above, this solution has the constraint that the VPN provider must manage the RTR Router within the CPE, a situation that may not be desirable.

4.2.1 IP VPN-aware correlation

An alternative solution requires adding a new feature to a Layer 2 monitoring system. This feature would allow the layer 2 device to be ***IP VPN-aware.*** This means that the monitoring system is capable of correlating IP VPN packets observed from different measuring points of layer 2 networks. The high level architecture of this measurement approach is shown in Figure 4. In Figure 4, the layer 2 networks could be Frame Relay networks or ATM networks, for example. It is assumed that there is no IP routing occurring in the layer 2 network. The IP network, which connects the layer 2 networks, includes IP routing and lower layer switching. This solution assumes that the measuring points are passive so that the entire IP traffic at points A and B are available at the new IP VPN-aware box (which does not currently exist, to the best of the authors' knowledge). This device is

required to measure the SLA metrics specified in Section 2, i.e. packet loss, throughput, delay/jitter, and availability.

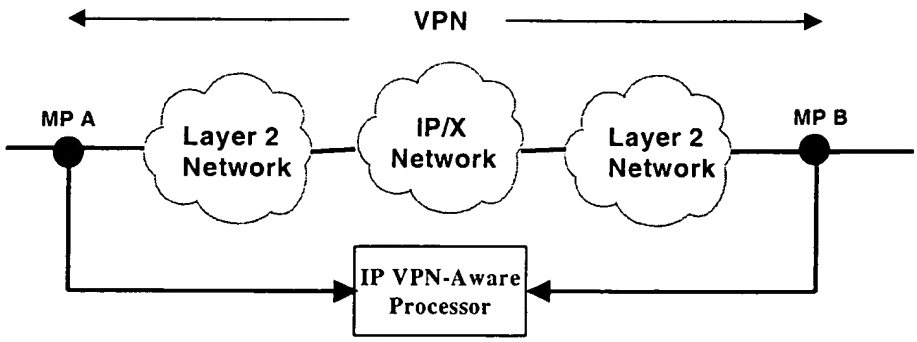

Figure 4: A network topology indicating an end-to-end VPN

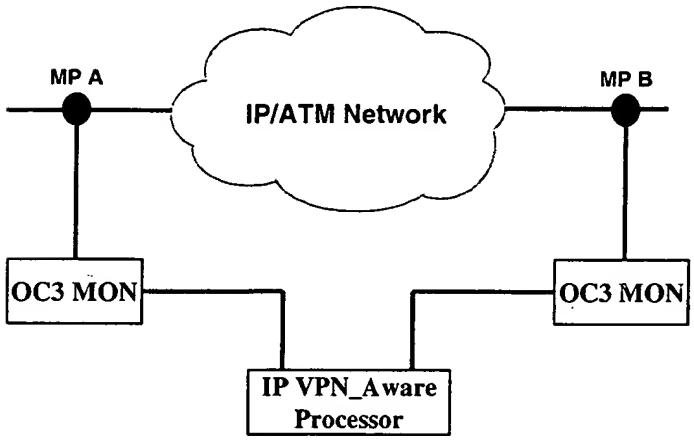

Figure 5: A network IP over ATM network with end-to-end measuring points

The IP VPN-aware box can be applied to any layer 2 technologies. For this discussion, we will use OC3MON as an example. The detailed architecture is shown in Figure 5. The key functions of the IP VPN-aware box are:

1. Identification of the VPN whose SLA metrics are to be measured
2. Synchronization and IP packet (Frame) alignment of the IP packets coming from the two measurement points A and B
3. Measurement of packet loss, throughput, delay, and jitter of the selected VPNs

4.2.2 VPN Identification

The VPN-aware box must extract information about the packets that are sent across individual IP VPNs. An IP VPN can be defined by IP source and destination (host) addresses, or IP source and destination network addresses. Using this definition, there could be thousands of VPNs supported through MPs A and B. It is also possible to define VPNs that carry only certain type of applications such as TCP or UDP only traffic. The IP VPN-aware processing should be programmed to handle any combination of these cases. In some cases, VPNs can also be identified by their virtual circuit identifiers in layer 2 (e.g., ATM PVCs).

4.2.3 Synchronization and Frame Alignment

Synchronization refers to the identification of the same IP packets observed from the two MPs of the IP VPN. This function is necessary for measuring packet loss, delay, and jitter. One way to achieve synchronization is to find some way to uniquely identify an IP packet. Conventional approach uses CRC calculated over the entire IP packet to provide a pseudo-unique framing pattern. Our proposed method does not require CRC computation and thus save a lot of processing. We propose to use a combination of the **source and destination IP addresses** (each 32 bits), the **IP Identifier** field (16 bits), the **Fragment flag** (3 bits), and the **Fragment Offset** field (13 bits) as a framing pattern for the unique identification of an IP packet within an VPN. The IP identifier is unique within the IP flow identified by the source and destination address if there is no fragmentation. If there is fragmentation along the VPN path (this is not common, but possible), the Fragment Offset field is needed to make the framing pattern unique. The 3-bit Fragment flag is there to make the frame alignment process simpler. If the Fragment flag is 000, there is no fragmentation and the fragment offset field will not be needed for frame alignment. With the framing pattern being unique for each packet of the VPN, it is conceptually easily to align the two observed VPN MPs. However, under close examination, one needs to solve the following issues:

1. Packets can arrive out of order or duplicated
2. The overhead data capacity needed to support this measurement
3. The frame alignment algorithm

Details of the complete packet alignment algorithm will be given in Appendix B. Here we show solutions to a few key issues.

Packets arriving out of order or duplicated

In IP, packets are allowed to arrive at the destination out of order or duplicated. A higher layer protocol such as TCP contains a sequence number and maintains a buffer of IP packets that is used to rearrange the out-of-order packets. Our approach solves the out-of-order or duplication problem differently: a key observation is that the packet headers sent from A to B via the overhead channel (the path over which monitoring information is transmitted) will not suffer loss packet or out-of-order degradation. Using these overhead packets as the gauge for framing makes us immune from the out-of-order problem. Again details of this procedure are provided in Appendix B.

Overhead capacity reduction

Overhead capacity refers to the traffic between OC3MON and the IP VPN-Aware Processor. A straightforward implementation is to take the IP headers generated by the OCnMON at MP A and to send them to the OCnMON at MP B for comparison and correlation. The problem of this approach is that a substantial amount of data will need to be sent for correlation. For example, if the average IP packet size is 800 bytes, and assuming only a portion of the IP header is required (~15 bytes), an OC3MON at 155Mb/s will generate overall data of about 3 Mb/s. This is a non-trivial addition of data capacity for just doing SLA monitoring! Certainly, a more efficient approach is needed.

In packet loss calculation, one can be satisfied with a packet loss count of say, once a minute (traditional data collection uses 15-30 minutes interval). At the end of each interval (i.e., one minute), we can just send a small number of IP headers (e.g., 15) along with some supplemental information to segment the IP packet stream into frames and to compute loss for each frame. This would reduce the overhead capacity to only a few bytes per second, a tremendous improvement!

Frame alignment problem

To achieve frame alignment between the two observed points, we start with the following algorithm:

1. Identify a framing pattern from MP A, say F(A1)
2. Starting from an arbitrary point at the IP header stream at MP B, say F(Bi), compare F(A1) with F(Bi), if equal, declare frame aligned, and note down time i. Alignment finished.
3. If comparison at step 2 is not equal, increment i, and go back to step 2.

This algorithm is simple but has a problem: when do we start the search on the IP stream at MP B? If we start too early, it will require a lot of storage of the IP packets from B and also waste a lot of processing time. On the other hand, if we start too late, we will miss the correct framing pattern at MP B. To have a robust and efficient frame alignment scheme, we propose to have MP A attach a local time stamp when the IP header information is sent to

MP B. The time stamp will be used to synchronize the clock at B and also give a reference point of where to start the searching. Of course, we must allow for the delay due to the IP network and also the inaccuracy of the clocks due to drifting and inherent inaccuracy of the clocks. However, those inaccuracies typically will be in the order of less than 50 ms. As a result, searching within a window size of 100ms will be good enough. This will not amount to excessive storage and processing.

The simple framing algorithm shown above does not deal with packets that arrive out of order. We present a more detailed algorithm that deals with this issue in Appendix B.

4.2.4 SLA Metric Calculation

Packet loss calculation:
As described above, the duplicated packets are accounted for in the examination of the IP framing patterns as the VPN is observed at B. Therefore, the IP VPN-aware processor is only concerned about lost packets. One can then designate a frame size of n packets, delineated by two framing patterns, F_1 and F_{n+2}, at MP A. When the same two framing patterns are observed at MP B with a packet count of m packets between them, the packet loss count of $(n\text{-}m)$ packets can then be computed. This high-level description of the packet loss measurement is illustrated in Figure 6.

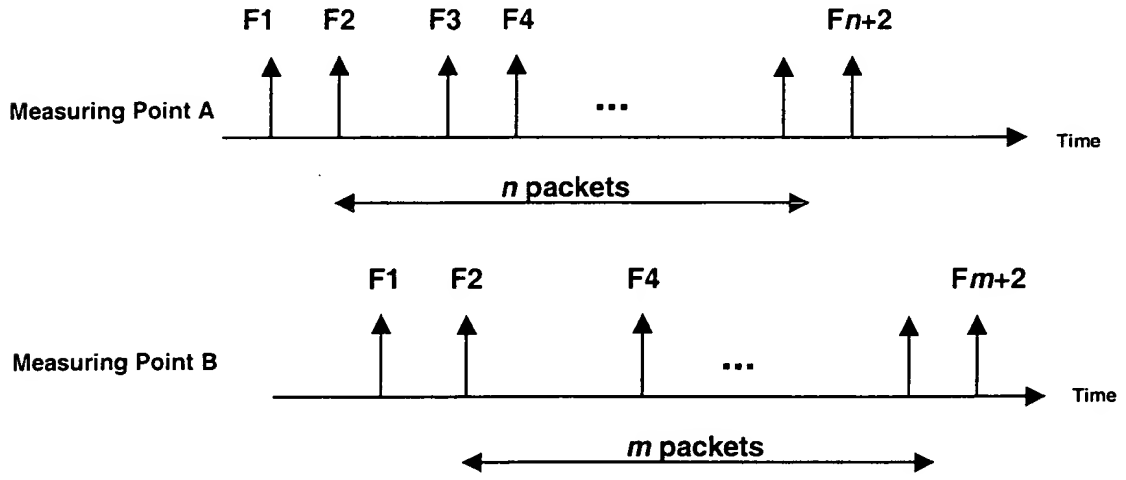

Figure 6: Measurement of Packet Loss

Delay and jitter measurement:
Referring to Figure 6, because every IP packet is associated with a time stamp local to the OCnMON, the difference between the time stamps corresponding to the same framing pattern at A and B gives the delay from A to B for that particular IP packet. This information can then be used to calculate average delay, peak delay, as well as jitter, which is defined to be variation in delay. Measurement of this one-way delay depends on the accuracy of the time stamps.

Throughput of a VPN:
Throughput can easily be measured by counting the number of packets at the exit point, i.e. at B, within certain timing window.

Availability of a VPN:
Availability can be computed as the percentage of the total time that the loss exceeds a given level. Note that the time resolution of the availability is limited by the frequency of the loss computation.

4.3 A "Sub-network" Approach

Each of the above two solutions, namely, CPE-based monitoring, and monitoring at the edge of the network with IP VPN-aware correlation has pros and cons. One clear disadvantage of the CPE-based approach is that one

monitoring device is needed per customer. When scaled to large number of CPE locations, the cost factor may present a significant barrier to its practical usage. The IP-aware approach will have similar economic problem if applied to each CPE access line. However, its usage in the core network, where large numbers of VPNs have been aggregated. is a viable approach. In this section we describe how various techniques can be combined to present an economically scalable solution.

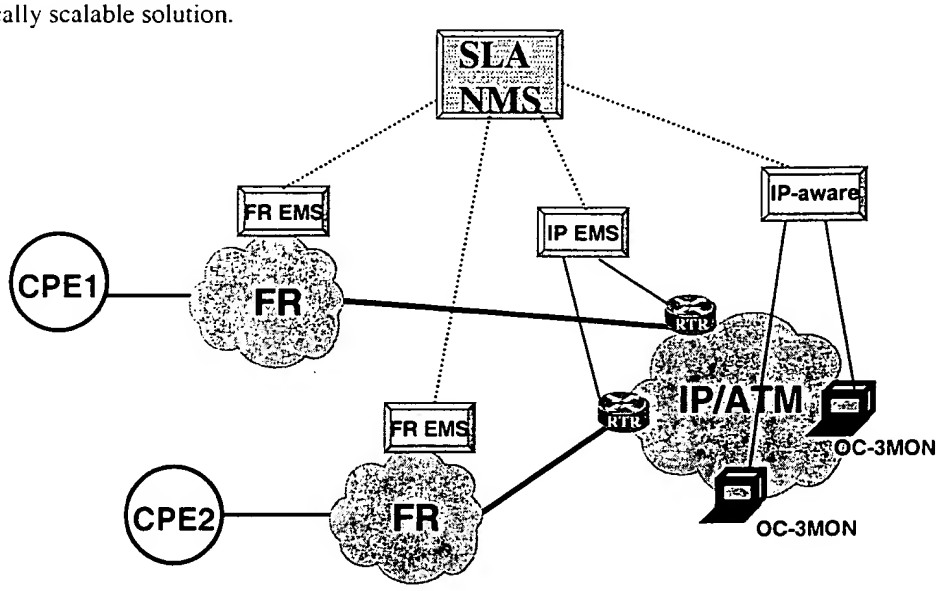

Figure 7: Subnetwork SLA approach

Observing that the difficulty of the cost scalability is in the access network, we will consider a solution that does not require monitoring per CPE. In our approach, we divide the end-to-end network into a number of "sub-networks" (not to be confused with the subnets in IP), corresponding to a particular technology. We then monitor each sub-network with whatever solution that is most suitable for that technology. The SLA monitoring results from each of the sub-network monitoring are then combined to construct the end-to-end SLA metric. This idea is illustrated in Figure 7. To implement this idea, we need to solve the following problems:

- Understand the mapping between different layers (e.g. Frame Relay and IP, ATM and IP, ...) with respect to SLA metrics
- Find an economical way to monitor SLA at the access network

4.3.1 Mapping between loss at the transport layer and the IP layer

ATM and Frame Relay switches collect various statistics about the PVCs they carry. However, the switches are not aware of the IP traffic being carried by the ATM cells or frame relay packets. From the ATM and Frame Relay loss statistics we would like to compute loss at the IP level for each PVC. In the Frame Relay case, one IP packet is carried by one Frame Relay packet, so the IP loss is equal to the Frame Relay loss.

The ATM case is more complicated because an IP packet may be carried over several ATM cells. The number of cells lost at the ATM level (which can be read using SNMP from MIBs in the switches) is an upper bound on the number of IP packets lost. From just ATM cell statistics, we cannot convert this upper bound on lost packets into a useful bound on the packet loss rate. However, at the adaptation layer above ATM, there is a one-to-one relationship between the AAL5 packets and the IP packets. Thus, dividing the cell loss by the number of AAL5 packets sent gives an upper bound on the IP packet loss rate.

It is also possible to detect a large portion of the IP packet loss by monitoring the AAL5 packets at the termination point of the PVC. Each AAL5 packet has a CRC field. If one or more of the ATM cells making up the AAL5 packet is lost, the computed CRC will not match the CRC field. In this case, the receiver will discard the AAL5 packet and the IP packet that it contains. The number of AAL5 packets that will be discarded divided by the number of AAL5 packets sent is a lower bound on the IP packet loss rate. In most cases this ratio is also a good estimate of the IP loss rate. However, when all of the cells making up an AAL5 packet are lost, the receiver can

not detect the loss; it will not know to compute the CRC since there is no trailer for the packet. For typical IP packet sizes (usually greater than 1 ATM cell payload) and typical ATM loss rates, such undetectable losses will be very rare. However, when there is a configuration problem or congestion causing larger amounts of loss, the estimate may be less accurate. A factor working in favor of this estimate is the fact that many PVCs will be multiplexed over the same ATM connection so loss of several consecutive ATM cells in one PVC is less likely.

4.3.2 Integrated Subnetwork Approach

Our solution focuses on a large family of network scenarios where a layer 2 technology such as Frame Relay or ATM makes up the access network. This scenario reflects the current network scenarios of packet service network providers including LECs, IXCs, and CLECs. In the layer 2 access network, we suggest that one can construct the SLA contribution with respect to the layer 2 network by polling relevant MIB information for the layer 2 switches. Polling switch MIBs is an economically scalable solution since the complexity is proportional to the number of switches, but not the number of access lines.

A number of issues will be addressed in the MIB-based approach:

- Autodiscovery of the virtual circuit
- Collection of per PVC statistics such as frame/cell loss or throughput and compute end-to-end PVC statistics from switch MIB information
- Delay measurement

The following describes in detail the proposed approach for Frame Relay. A similar approach for ATM was discussed in [PerfPoint99].

4.3.3 SLA Parameters associated with the Frame Relay Subnetwork

The section will provide the SLA defined parameters associated with the Frame Relay switches.

4.3.3.1 Utilization

This SLA parameter is defined as "*The percentage of the customer allocated PVC capacity used.*" The measurement associated with this parameter is defined as:

$$PVC_{Util}(\%) = \frac{UtilizedBandwidth}{AllocatedBandwidth} \times 100$$

Equation 1: Frame Relay PVC Utilization

Where the Allocated Bandwidth is the Frame Relay parameter CIR (Committed Information Rate). In the case of Frame Relay service, utilization can be greater than 100% if the PVC is set up with an excess burst rate (Be) grater than zero.

4.3.3.2 Packet Loss Ratio

Packet Loss Ratio (PLR) is defined as "*The percentage of the total number of lost frames*", on a given Frame Relay connection and is given by:

$$CLR_{FR}(\%) = \frac{N_I Packets - N_E Packets}{N_I Packets} \times 100$$

Equation 2: Frame Relay Packet Loss Ratio

Where $N_I Packets$ is the Ingress packet count and $N_E Packet$ is the Egress packet count.

4.3.3.3 PVC Availability

This parameter is defined as "*The percentage of time a given PVC is operational*" and can be quantified as:

$$PVC_{Avail}(\%) = \frac{ElapsedTime - OutageTime}{ElapsedTime} \times 100$$

Equation 3: Frame Relay PVC Availability

The total elapsed time must be variable and based on specific customer requirements. The PVC outage time is based on the operational status of the entire PVC plus the sum of the time intervals during which the packet loss ratio exceeded the SLA loss parameter.

4.3.3.4 Delay

This parameter is defined as *"the amount of latency for packets to be carried through Frame Relay network."* This is an end-to-end measurement, which reflects the transit time across a service provider Frame Relay network.[4]

4.3.4 Data collection agent functional requirements

This section will provide the functional requirements for the data collection Operations System in support of the SLA parameters defined in Section 5.

4.3.4.1 Interface Requirements

The interface between the Network Management Layer (the data collection agent) and either the EML (Element Management Layer) or the NE (Network Element) must be defined in an AAL (Application to Application Language) between the development organization and the customer.

4.3.4.2 Frame Relay MIB Objects

The relevant MIB objects associated with Frame Relay Switches are based on the Cascade switch MIB: Cascade Communications STDX/B-STDX MIB definitions. It is almost certain that analogous variables are available from other Frame Relay switches.

4.3.4.3 Discovering the Routing of PVCs in a Frame Relay Network

SLA metrics apply to an entire PVC. In order to compute SLA metrics it is necessary to know the route of each of the PVCs. The element manager can discover the complete set of PVCs in a Frame Relay network by consulting the connection tables of the switches. The algorithm that follows is due to K. R. Krishnan [Krishnan96]; a few changes and corrections have been made. The algorithm makes use of a few MIB variables the definitions of which are quoted from the MIB document for Cascade switches.

cktSrcIfIndex = The ifIndex (interface-index) value of the corresponding [table] entry. [This is the first cktTable index]

cktSrcDlci = The DLCI used as the key for the circuit [connection]. For local DLCI significance, this is the local DLCI. For global DLCI significance, this is the remote DLCI. [This is the second cktTable index]

cktDestDlci = The DLCI which is the destination of the key DLCI. For local DLCI significance, this is the remote DLCI since the key DLCI is the local DLCI. For global significance, this is the local DLCI since the key DLCI is the remote DLCI.

cktDestNodeId = The destination node ID for this circuit [connection].

cktPath = The circuit path consisting of a sequence of outbound interface indexes along the established circuit. The format is interface:interface:interface:

We follow the same procedure at each switch; call it Switch A. We look at the connection table called cktTable, which has two indices, cktSrcifIndex and cktSrcDlci. For each cktTable entry determine the next switch, call it switch B, from the variable cktDestNodeId and obtain its route by means of the variable cktPath. However, in

[4] In a network topology that implements router/ATM combination, delay can be measured using "ping" features in the IP routers.

order not to process the same connection twice when working on switch B, we do the following. For each switch k, and for each interface ifind, define an array DUPL(k,ifind) in which we maintain a list of DLCIs that were already processed at the switch at the other end. All the arrays DUPL(k,ifind) are initialized so that they contain no valid DLCIs to begin with, for example –1.

The algorithm proceeds as follows:

For each switch k = 1,...,n. for each value of ifinda = cktSrcIfIndex, for each value of index = dckSrcDlci

Determine cktDestNodeId(k,ifinda,index)=switch j

Determine the route for this connection from cktPath(ifinda, index); in particular, we obtain the port index, ifindb, of this PVC at the next switch j.

Obtain cktSrcDlci(k,ifinda,index)=dlca, if dlca is already in the array DUPL(k, ifinda), skip to the next value of index; otherwise do next step

Obtain cktDestDlci(k, ifinda, index) = dlcb, add the entry dlcb in DUPL(j, ifindb) corresponding to the next switch j for this connection.

Load a structure that specifies how to follow this PVC from switch k to the next switch with nodeId = j
FollowPVC_struct(k, input DLCI at k, input port at k, output DLCI, input port next switch, next switch Id)
input DLCI at k = index
input port at k = ifinda
output DLCI = input DLCI at j = dlcb
input port next switch = ifindb
next switch Id = j

Go to next cktTable entry

For example, we can use the set of structures FollowPVC_struct to determine the number of frames lost over a period of time for a PVC (see subsection on Frame Relay Packet loss ratio a few paragraphs below). We start at the ingress point of a PVC, collect the relevant MIB variables at the ingress switch, and use the set of structures to follow the connection to the next switch. The structure has all the necessary information to uniquely specify the required row of the cktTable at each switch; we now collect the relevant MIB variables from this switch; continue until the egress point is reached.

4.3.4.4 SLA Measurements

This section provides the measurement requirements for the SLA parameters defined above.

- *Frame Relay PVC Utilization*

This measurement is the ratio of the utilized bandwidth to the allocated bandwidth of the PVC based upon the traffic flowing for a period of time divided by the Committed Information Rate (CIR). This is an average for the time period determined by the service provider in the granularity of the report and must be available for each direction of a PVC. Thus the average utilization is:

$$\overline{Utilization} = \frac{BitsTransferred}{CIR \times \delta t}$$

Equation 4: Frame Relay PVC Utilization

Where CIR is the Committed Information Rate for a given δt time period. The number of bits transferred over a period of time is obtainable from the MIB variable cktOutOctets.

- *Frame Relay Packet Loss Ratio*

The total number of frames lost for a PVC in a Frame Relay switch is obtainable from the following combination of MIB variables: (cktInFrames – cktOutFrames + cktOutLostFrames).

End-to-End packet losses due to sum total of all losses spanning multiple Frame Relay switches.

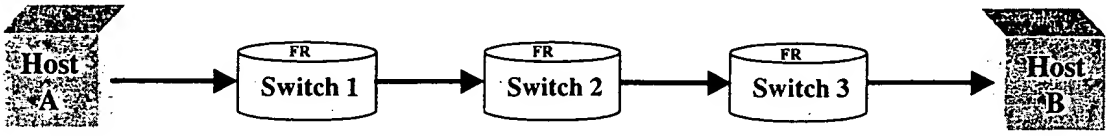

Figure 8: A network topology representation of Frame Relay Switches in tandem

The following formulation represents the PLR_{FR} for the above tandem Frame Relay switch topology.

$$PLR_{FR} = \frac{loss(S1) + loss(S2) + loss(S3)}{Frames(inS1)}$$

Equation 5: Frame Loss Ratio for a set of Frame Relay switches in tandem

Loss is obtainable from the MIB of each switch:
loss = (cktInFrames – cktOutFrames + cktOutLostFrames) – [cktInDiscards]
and the ingress frame count to the PVC is obtained from cktInFrames at the first switch:
Frames = cktInFrames

- *Mapping of Frame Relay Packet loss to IP packet loss*

We have examined three operating Telcordia routers and found that the MIB variable ifMtu is 1500 bytes for all of their Frame Relay interfaces. Each of these routers had one or more such interfaces. The definition of ifMtu is: "The size of the largest datagram which can be sent/received on the interface specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface." Since the maximum payload size of Ethernet is 1500 bytes, we assume that no IP datagram is fragmented to be transported on the Frame Relay network. Thus, the PLR is one-to-one between Frame Relay and IP.

- *Frame Relay PVC Availability*

Using Equation 3, we specify an elapsed time. The outage time is the sum of the time intervals during which the PLR exceeded SLA specifications plus the time during which the operational status of the entire PVC (MIB variable cktOperStatus) was not active.

- *End to End Frame Delay for a PVC*

Frame Relay switches measure three delay quantities through the use of OAM frames to the next switch and on a per PVC basis:

cktRtMinDelay The minimum round trip delay
cktTtMaxDelay The maximum round trip delay
cktRtAvgDelay The average round trip delay

For a connection going through a number of switches it is only necessary to add the corresponding quantity for each switch. We cannot obtain, solely from MIB variables, the one way delay. However when long delays are experienced it is usually one way that contributes most of the delay. The round trip delay measurement is sufficient for SLA.

4.3.5 The Combined Solution

The core network is assumed to be an IP network. Thus many existing IP monitoring tools can be used. Examples include Cisco's RTR, Surveyor, the PANX tool, or the IP-aware correlator. (See Appendix A for descriptions.) The key here is to have a north bound interface to the SLA NMS.

The SLA NMS combines all the SLA metrics information collected from individual sub-network EMSs and reconstructs the end-to-end SLA for a particular VPN.

5 POSITIONING TELCORDIA SLA MANAGEMENT OSS

5.1 Current Market Scenario

Currently there are dozens of products or tools in the market, which claim to support SLA. Market leaders include Visual Network's Visual UpTime, Concord's Network Health, and Cisco's RTR. In the Frame Relay SLA market, Visual UpTime seems to be the leader, with AT&T and a number of RBOCs evaluating this product. Visual UpTime also begins to offer ATM SLA solution, based on the same approach as their Frame Relay product, i.e. a passive monitoring device capturing ATM level parameters. Visual UpTime is also capable of displaying statistics on the IP level, such as traffic mix (TCP, UDP). However, as mentioned before, it is currently not capable of correlating IP VPN at different MPs. In other words, Visual UpTime alone will not give end-to-end SLA for an IP VPN.

Another significant shortcoming of Visual UpTime is that it does not scale well economically. Each Visual UpTime may cost a few thousand dollars for just first installed cost. This will be hard to justify for lower end customers.

5.2 Strategy and recommendation

Currently, Telcordia's SLA Manager (SLAM) is implemented as a feature in Performance Point, which can use DCOS-2000 for collecting SLA metrics. As the requirement is written [PerfPoint99], SLAM is capable of performing simple mathematical operations on collected variables, a capability well matched to the MIB-based approach. Based on what have today, it should not be difficult to extend the SLA to support polling of Frame Relay MIBs and compute Frame Relay SLA metrics. This should put us in a good position to offer a scalable Frame Relay solution that product like Visual UpTime is lacking.

In the next step, it would be strategically important to develop capability to interface to other data collection products such as RTR. Our product should also have the function of integrating data collected from different monitors and display the integrated SLA view.

We should concurrently pursue an implementation of the IP VPN-aware correlation function. We should first build a prototype to demonstrate the feasibility, followed by a thin requirement description of the functionality. Seeking external partners is also a potential option.

6 CONCLUSION

In this document we provide a solution to the SLA measurement problem by examining different schemes and methodologies. The key results are:

- Many Telcos are currently thinking about offering IP VPN over various layered 2 technologies including Frame Relay, ATM, IP over SONET... They need to solve the problem of SLA assurance in additional to SLA monitoring and data collection. Currently, partial solutions exist but are not capable of satisfying all the requirements of cost-effectiveness, accuracy, and integration with existing and new OSSs. The addition of IP equipment in the telco's network means that they must manage IP equipment.
- Survey of industry with respect to SLA monitoring schemes and a gap analysis on what is still missing. After a survey of what exists in the SLA market, it is recognized that the existing solutions are not cost scalable; this is because existing monitoring approaches require one

monitoring system per access line. While this is acceptable for monitoring large pipes such as those found in the core network, it may not be acceptable in the access network where there can be thousands of access lines per central office. In this document, we propose a scalable solution that involves an integrated NMS and MIB-based EMS for the access network. Details of the access EMS for Frame Relay are also given.

- While examining the passive monitoring approach, we observed that IP-VPN-awareness is a key function that is missing in existing products. We propose new algorithms and design for a new box that provides this IP-VPN-aware function. We obtained a very interesting result that is potentially patentable.
- Finally, we made a list of recommendations on the direction of the Telcordia SLA OSS products.

APPENDIX A. EXISTING MONITORING TECHNOLOGIES.

In this section we will provide a list of technologies (WAN Service Level Reporting Products) and their corresponding features.

- **Visual UpTime**

Visual UpTime from Visual Networks is a passive monitoring system that supports Frame Relay and ATM SLAs. In Frame Relay, Visual UpTime taps into the network through a T1 monitoring jack. The Frame Relay signal is then captured in the Analysis Service Element (ASE), which is basically an embedded SLA device integrated into the CSU/DSU. The ASE collects and stores the captured data and has a 10BaseT interface to a Management Console, which runs on a PC platform and is responsible for performing analysis of collected data and produce SLA reports. Visual Network purchased Net2Net and uses their ATM analysis tool for the ATM monitoring part. There are currently less detailed information on the ATM version of Visual UpTime, but it is expected to use OCnMON device for passive data capture and writing SLA reports similar to that of Frame Relay SLA. In addition, a different module called Internet ASE supports IP level traffic statistics such as FTP, Telnet, ICMP, DNS, BOOTP utilizations. However, no correlation of IP information at different points of the VPN is performed.

Visual UpTime is a leading product in SLA management. Various network providers are currently evaluating Visual UpTime. Sprint Corp. is a reseller of Visual UpTime. MCI Communication is both a reseller and a user. Bell Atlantic Network Integration is also a reseller of the product. However, there are concerns about scalability cost since for each Frame Relay access line a monitoring system is needed.

- **Surveyor:**

Advanced Network & Services' Surveyor project [Surveyor99] uses an active monitoring approach to make accurate delay and loss measurements. Surveyor boxes, customized PCs, are placed at various places around the Internet (currently in 42 locations around the world, mostly at university and government sites) and send time-stamped messages to each other. The boxes are kept closely synchronized using Global Positioning System (GPS) receivers; delay measurements are accurate within 50 microseconds. Currently, measurements are taken continuously at a rate of approximately two measurements per second between each pair of hosts. Each Surveyor box computes the delays and losses for each path for which it is the destination. The data is stored in a huge database at Advanced Network & Services. Jitter can be computed from the delay measurements. The paths between the Surveyor boxes are measured every few minutes using the traceroute program. The Surveyor project implements the performance metrics proposed by the IP Performance Measurement (IPPM) working group. There is a similar project by RIPE-NCC in Europe which is also implementing the same performance measurements using a different "Test Box" with GPS receivers.

Projects like Surveyor and RIPE-NCC's Test Box use active monitoring to get *samples* of some of the SLA metrics: delay, loss, and jitter.

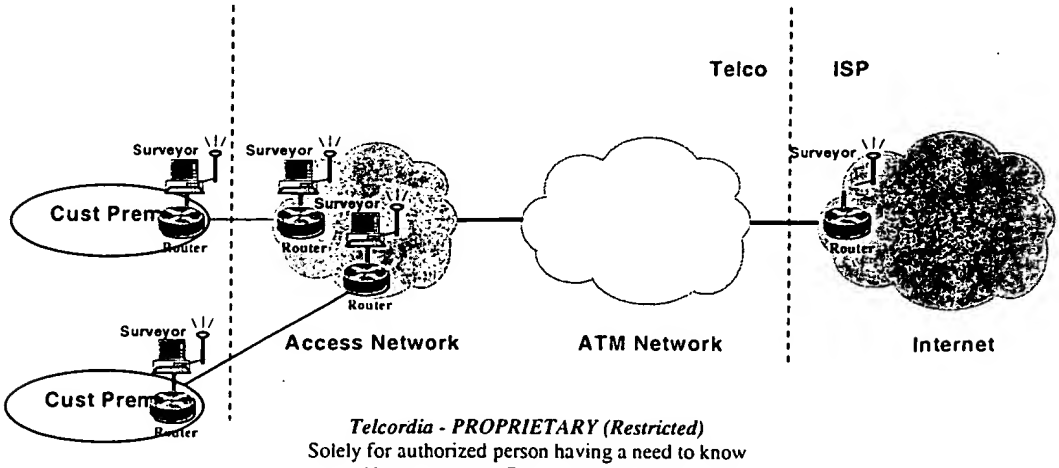

Figure 9: A network topology utilizing Surveyor for SLA parameter measurement

- **RTR:**

The RTR (Response Time Reporter) is a component of IOS on Cisco routers, where source router IOS agent periodically creates application-aware synthetic transactions to closely emulate applications' performance on network. RTR uses TRACEROUTE to determine active IP paths and response times for each hop and requires RTR responder in target router. Measurement can be accessed using SNMP interface to reporting applications. The SLA parameters such as delay, packet/cell loss, inter-packet delay variation are supported, while throughput is planned for the future releases. Configurable polling intervals and user-customizable thresholds generate SNMP traps upon SLA violation and trigger a path echo probe from hop-by-hop analyses.

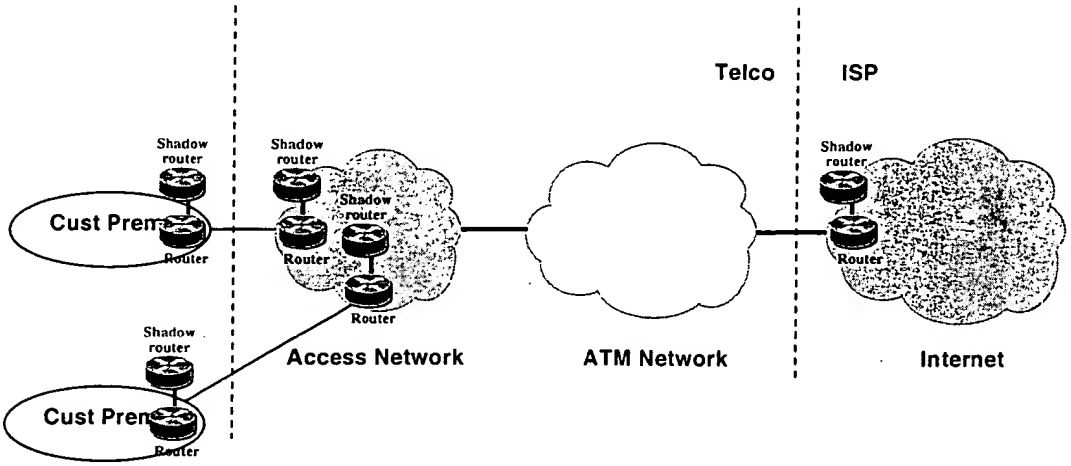

Figure 10: A network topology utilizing RTR routers for SLA parameter measurement

- **PANX:**

Panxtool is a GUI-based tool, which runs three distinct performance tests: *file transfer delay*, *packet loss rate*, and *throughput* as specified in the AIAG document.[5] It operates as a client program under the classic client/server paradigm. The server is the TCP discard server, which merely performs passive opens of TCP connections on the well-known TCP port number 9. All data received at the server is acknowledged via TCP, only to be discarded by the server process. The server doesn't perform active closes; this is a function of the client process.

Panxtool, which performs active TCP opens and closes to the server, is responsible for all performance measurement tasks and data logging. It employs the services of the anx monitor/pseudo device driver developed for the FreeBSD kernel. The anx device driver "listens to" TCP traffic departing and arriving at the client machine. The device driver provides total IP packet and byte (inclusive of IP and TCP overheads) counts in both directions of a duplex channel specified by source address, source port, destination address, and destination port. Any of the four parameters can be wild-carded (i.e., match anything). The device driver also counts all TCP flags (ACK, RST, SYN, FIN, PUSH, and URG), as well as counts the number of outgoing TCP retransmitted segments and bytes. The device provides a trace of all packets monitored. The trace records the time (second resolution), TCP flags, packet size, direction, and retransmission indicator for each monitored packet.

- **OC3MON:**

[5] Metrics, Criteria, and Measurement Requirements for ANX Release 1, Issue 1," pp. 47-53.

OC3MON is a passive monitor for ATM over OC3. It works by splitting a small amount of optical power from an OC3 cable, and streaming the ATM cell data into a pair of Fore ATM cards in a Free BSD-based host (one card for each direction of traffic). The OC3MON software captures or analyzes the cell streams in real-time as shown in figure below. Both the software and source code is freely distributed, allowing modification for specific data-processing purposes (such as SLA). The stock software that comes with OC3MON can operate in two modes: raw capture, and flow analysis. In raw capture mode, a configurable number of leading cells from each AAL-5 frame are dumped into local RAM, until the RAM cache is full. This data can then be offloaded to a remote analysis machine. This method is provides non-contiguous information on the cell stream. Of more interest is the real-time analysis mode; however, this built-in processing functionality is oriented towards traffic statistics collection rather than SLA metric measurement. In this mode, a variety of specific and aggregate data is collected about flows and collections of flows—where the definition of a "flow". The statistics collected include application-specific data (e.g. percent of packets which are http) and throughput data (e.g. top ten highest packet percentage flows).

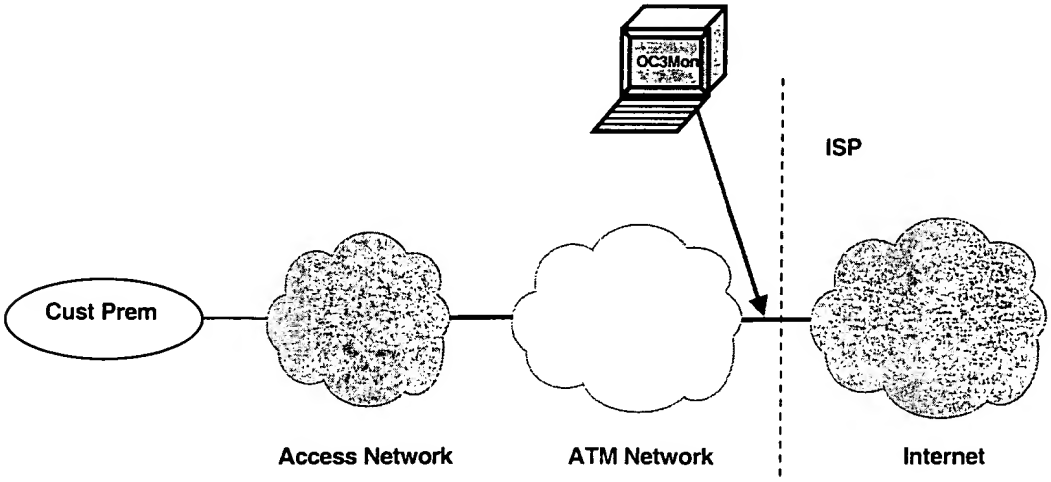

Figure 11: A network topology utilizing OC3MON for SLA parameter measurement

Dag [WAND99] is a similar passive monitoring tool being developed at the University of Waikato. A custom PCI adapter card reads data from a split OC3 cable, as with OC3MON. Because the hardware is customized, Dag has the additional capability of calculating CRCs for each cell in hardware, as well as attaching to each CRC a timestamp in hardware from a GPS-based time source. Additionally, the focus of the software being developed around the Dag card is on performance measurement, which makes it more directly applicable to SLA metric measurement. As with OC3MON, source code used in the Dag experiments is freely available. Of particular interest is the IP packet delay experiments, which utilize two Dag at either end of the network, both of which are connected to a central processor. The central processor using CRCs from each Dag uniquely correlates the cells; combined with the GPS timestamp data, it produces comprehensive and accurate delay statistics.

❑ **Polling MIB:**

Polling MIBs, as seen in Figure 12 below, is a non-intrusive means of extracting information from the network elements associated with SLA parameters. This can be achieved via SNMP messages either directly accessing the network element or the element management system. MIB provides information about what the NE sees associated with the traffic, error, and status of the interface. However, there are limitations using this method. Delay and Jitter can not be measured using MIB objects. Additionally, for IP routers, packet loss can only be determined in a statistical manner. Finally, there is some concern regarding the accuracy of IP MIBs.

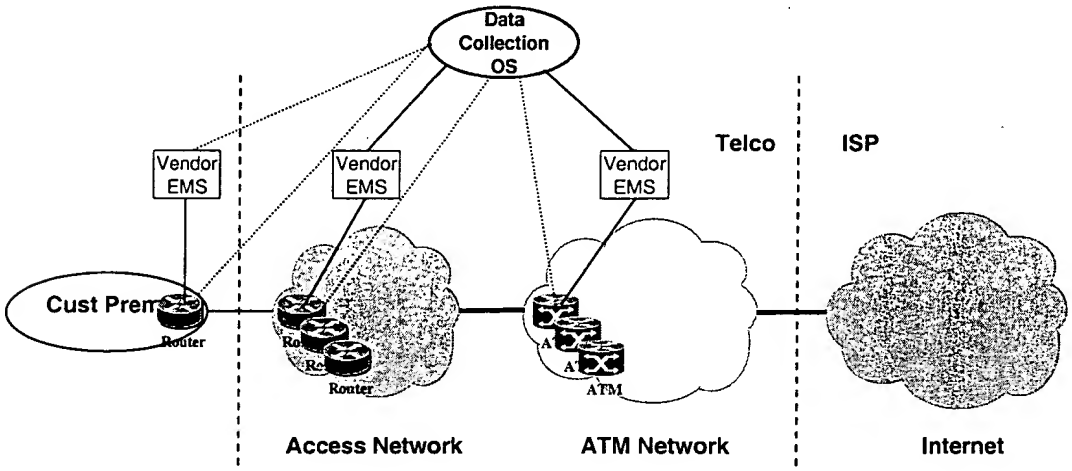

Figure 12: A network topology utilizing MIB Polling capabilities

❑ **RMON:**

The RMON specification (Standard in 1992 as RFC 1271 and subsequent RFC 1757) provides network administrators with comprehensive network fault diagnosis, planning, and performance tuning information. It delivers this information in nine groups of monitoring elements, each providing specific set of data to meet common network monitoring requirements. Each group is optional and vendors do not support all the groups within the MIB. Table 1 provides a list of the nine groups and their associated functions.

Capabilities:

The deployment of RMON can provide the SLA parameters such as *Availability*, and *Throughput*. The throughput can be measured at a single point by checking a table that provides the number of packets that have gone through the network. However, RMON cannot measure loss rate and delay/jitter unless some new extension such as the one described in Section 4.2 of this document is used.

Statistics Group	Contains statistics measured by the probe for each monitored interface on this device.
History Group	Records periodic statistical samples from a network and stores them for later retrieval.
Alarm Group	Periodically takes statistical sample from variables in the probe and compare them with previously configured thresholds. If the monitored variable crosses a threshold, an event is generated. A hysteresis mechanism is implemented to limit the generation of alarms. This group includes the "alarmTable" and requires the implementation of the "event" group.
Host Group	Contains statistics associated with each host discovered on the network.
HostTopN Group	Prepares tables that describe the hosts that top a list ordered by one of their statistics over an interval specified by the management station. Thus, these statistics are rate based.
Matrix Group	Stores statistics for conversations between sets of two addresses. As the device detects a new conversation, it creates a new entry in its tables.
Filter Group	Allows packets to be matched by a filter equation. These packets from a data stream that may be captured or may generate events.
Packet Capture Group	Allows packets to be captured after they flow through a channel.
Event Group	Controls the generation and notification of events from this device.

Table 1: Nine RMON groups and their associated functions.

APPENDIX B. IP-AWARE CORRELATION ALGORITHM.

Introduction

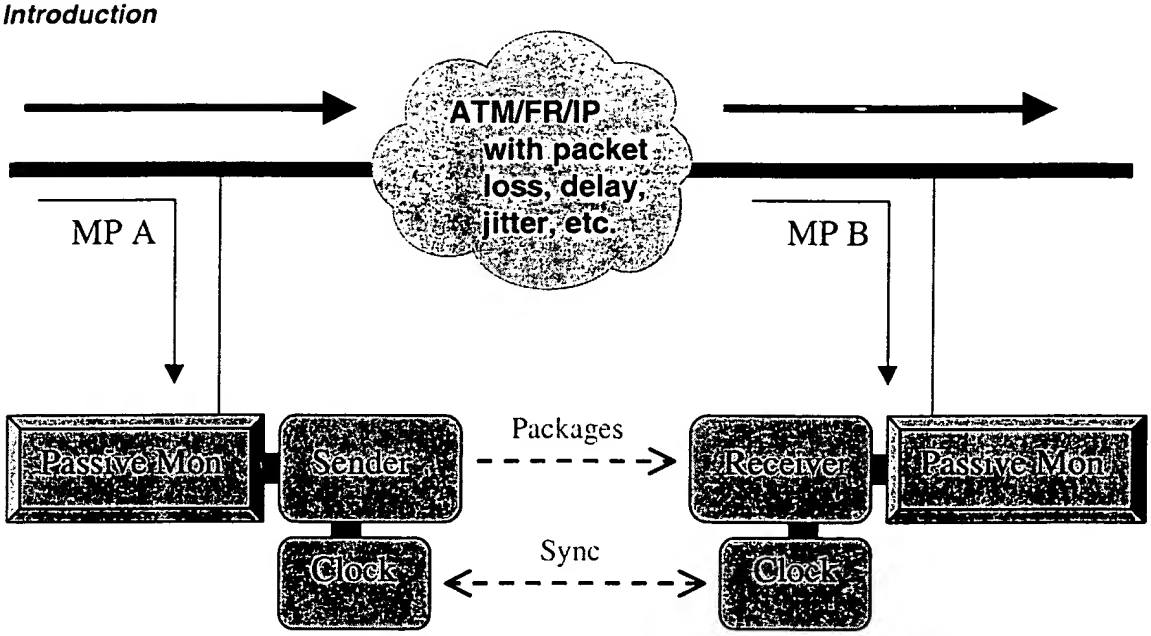

Figure 13: IPACA Architecture.

The IP-Aware Correlation Algorithm (IPACA) synchronizes the inherently error-prone, out-of-order IP packet-traces between measuring points in an ATM/FR/IP network. This synchronization is useful for accurately calculating a variety of SLA metrics at the IP level (e.g. packet loss, delay, jitter), as well as for collecting useful information for other purposes (e.g. network debugging, SLA assurance). The algorithm works with one direction of traffic flow—a similar mechanism may be applied to synchronize traces in the opposite direction, using the same measuring points. Network traffic flows past MP A, through the ATM/FR/IP network, and out through MP B. IPACA utilizes passive monitors at two measuring choke-points at each end of the network; it can use any passive monitoring hardware that is able to capture all IP headers flowing through the high bandwidth data-pipes at MPs A and B. The passive monitors at the measuring points are attached to (possibly auxiliary) computers that implement the algorithms described below. This description applies to the case of a single IP flow carried by the network. IPACA can be easily scaled to multiple VPNs or any other flow types by creating multiple instances of the same algorithm, as long as the VPNs or flows can be identified from the packet-trace (using IP addresses in the packet headers, layer-2 PVC ID's, etc). IPACA also uses a mechanism to synchronize time between the two MPs; this synchronization can come from an external source (e.g. GPS, NTP), or an internal network-based synchronizing procedure (using local free-running clocks).

Definitions

The term *packet-trace* refers to the serial traffic of IP packets flowing past a measuring point. A *frame* is a contiguous sequence of packets from an associated packet-trace. The goal of this algorithm is to synchronize frames associated with two packet-traces, namely the A-trace and the B-trace. Adjacent frames are not separated by any packets; i.e., all packets are contained within a frame. The sending workstation at MP A takes the first s packets from each frame and puts them (along with other data, as described below) in a *package*. This package is sent to the receiving workstation at MP B, where it is processed according to the algorithm described below. Associated with each frame is a *frame size* (*FS*), which is simply the number of packets contained in the frame. Each packet is associated with a *time-stamp* (*TS*) which is the time that the packet arrives at the MP.

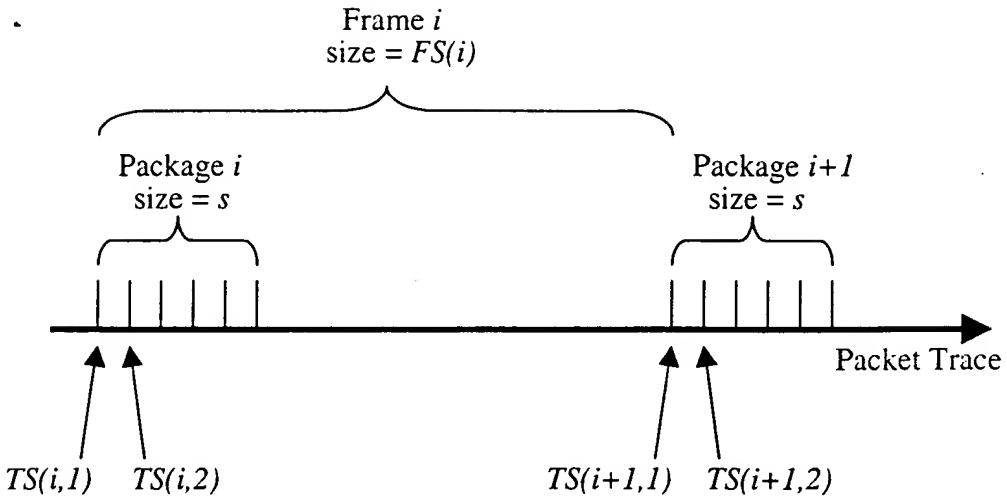

Figure 14: Definition of Terms.

Data Requirements

Both sender and receiver must independently maintain *header-storage* data structures, which store copies of IP headers as they pass by the MP. The *package* contains the information needed to synchronize the frames between A and B.

Package Data Structure

VPN identifier				
Flow identifier (identifies a flow within a VPN, e.g., a VPL or a socket connection)				
Frame number (for frame preceding the current package)				
Frame size (packets from start of previous package to start of current package) = *FS*				
Number of duplicate packets since start of previous package				
Package size (number of packet headers in current package) = *PS*				
Src. addr/port 1	Dest. addr/port 1	IP identifier 1	Frag. flag/offset 1	Time stamp 1
Src. addr/port 2	Dest. addr/port 2	IP identifier 2	Frag. flag/offset 2	Time stamp 2
Src. addr/port 3	Dest. addr/port 3	IP identifier 3	Frag. flag/offset 3	Time stamp 3
...	...	...	...	...
Src. addr/port *PS*	Dest. addr/port *PS*	IP identifier *PS*	Frag. flag/offset *PS*	Time stamp *PS*

Figure 15: Package data structure.

Most of the fields of the package data structure are 16 or 32 bit values. The addr/port fields are each a combination of an IP address (32 bits) and a port number (16 bits). The fragmentation flag and offset total 16 bits. The time stamp field may need to be greater than 32 bits to accommodate microsecond measurements.

The VPN identifier identifies the virtual private network from the customer's point of view; the VPN may connect several CPE locations. The flow identifier field identifies the particular pair of locations for which the SLA statistics are being calculated using the package (e.g., the endpoints of a virtual private line, or for more detailed statistics, the two IP addresses that are communicating, or the IP addresses and sockets that are connected).

Header-Storage Data Structure

VPN identifier				
Flow identifier (identifies a flow within a VPN, e.g., a VPL or a socket connection)				
Last frame number (frame number for most recently received package)				
Packet index for beginning of current frame = *FB*				
Current packet index (following most recently recorded packet) = *FC*				
Number of duplicate packets since start of previous package (may have to be stored per packet)				
Src. addr/port *FB*	Dest. addr/port *FB*	IP identifier *FB*	Frag. flag/offset *FB*	Time stamp *FB*
Src. addr/port *FB*+1	Dest. addr/port *FB*+1	IP identifier *FB*+1	Frag. flag/offset *FB*+1	Time stamp *FB*+1
Src. addr/port *FB*+2	Dest. addr/port *FB*+2	IP identifier *FB*+2	Frag. flag/offset *FB*+2	Time stamp *FB*+2
...	...	...	...	...
Src. addr/port *FC-1*	Dest. addr/port *FC-1*	IP identifier *FC-1*	Frag. flag/offset *FC-1*	Time stamp *FC-1*

Figure 16: Header storage data structure on the receiver monitor (MP B).

The Framing Algorithm

Workstations at A and B independently maintain header-storage data structures, which automatically store information on duplicated packets. The sending workstation at MP A periodically sends packages to the receiving workstation at MP B, using a reliable transport mechanism such as TCP. One package is sent at the beginning of each frame. When B receives a package, it searches through its header-storage to find a match with any of the headers in the package. This search is made more efficient by using (1) the synchronized time-stamps provided by the package as a lower bound on the time to begin searching in the header-storage, and/or (2) the FS of the previous frame, to guess the approximate location of the match. The latter cannot be used to synchronize the first frame, since in this case no previous frame has been synchronized. The first match is used as the start of the next frame. If none of the *s* packet headers in the package can be found in B's header-storage, the package is disregarded and B waits for the next package to arrive. This case is treated just as the normal case, except the current FS is added to the next FS when calculating SLA metrics (i.e. the current and next frame are treated as one big frame); this logic can be extended to multiple sequential occurrences of the "all headers missing" case.

The high level algorithm for identifying the frame boundaries on the receiver is shown in Figure 17. The monitor at MP B, the receiver, is continuously recording header information about all the packets that it receives. Each time the MP B monitor receives a package from the MP A monitor, it compares the headers in the package to the headers in its storage buffer. The receiver is in one of two states with respect to framing: it is *in frame* if it was able to match a recent package with its header storage; or it is *out of frame* if it has not received any prior packages or has not been able to match any recent packages. When the receiver is out of frame (lines 2-16 in Figure 17), it must identify the window within its header storage that the current package must be contained in based on the time stamp for the first header in the package and the time that the package was received. (See Figure 18.) This must be a large window because of the uncertainty in the monitor clocks and in the delay between the monitoring points. When the receiver is in frame, a smaller window can be identified based on the frame size (the number of packets sent between packages) provided in the packet. If one of the headers in the package is at the expected location in the header storage, then a small window can be searched for the start of the frame (lines 19-34). (See Figure 19.) Since the packet headers in the package are from consecutive packets, they should not be too far from each other in the header storage. If there is no header match at the expected location, a larger window must be searched (lines 35-56).

```
When receive a new package...
If not in frame
    Set window to search in header storage
        Start of window is based on the timestamp for the first header in the package.
        End of window is based on the arrival time of the package.
    Find first header in the window that matches any of the headers in the package.
    If no match
        Then still out of frame
        Abort and wait for next package.
    Else
        Set start of new frame to location of matching header in header storage.
        Record packet count for start of new frame.
        frameoffset = 0
        badframes = 0
        In frame.
    Endif
Else [already in frame from last package]
    Set expected frame boundary to recorded packet count + framesize + frameoffset in current package.
    If packet header at location indexed by frame boundary matches any of headers in package
        Set window to search in header storage
            Start of window is expected frame boundary minus SMALLSEARCH headers
                Where SMALLSEARCH is based on the packet count for the package
                and the amount of loss or reordering that is considered likely.
            End of window is expected frame boundary minus 1.
        Find first header in the window that matches any of the headers in the package.
        If no match
            Set start of new frame to expected frame boundary
        Else
            Set start of new frame to location of matching header in header storage.
        Endif
        Record packet count for start of new frame.
        frameoffset = 0
        badframes = 0
        In frame.
    Else
        Set window to search in header storage
            Start of window is expected frame boundary minus LARGESEARCH headers
                Where LARGESEARCH is several times SMALLSEARCH.
            End of window is expected frame boundary minus 1.
        Find first header in the window that matches any of the headers in the package.
        If no match
            badframes = badframes + 1
            If badframes > threshold (e.g., 3)
                Out of frame
            Else
                frameoffset = frameoffset + framesize
                Still in frame
            Endif
            Abort and wait for next package.
        Endif
        Set start of new frame to location of matching header in header storage.
        Record packet count for start of new frame.
        frameoffset = 0
        badframes = 0
        In frame.
    Endif
Endif
```

Figure 17: Framing algorithm.

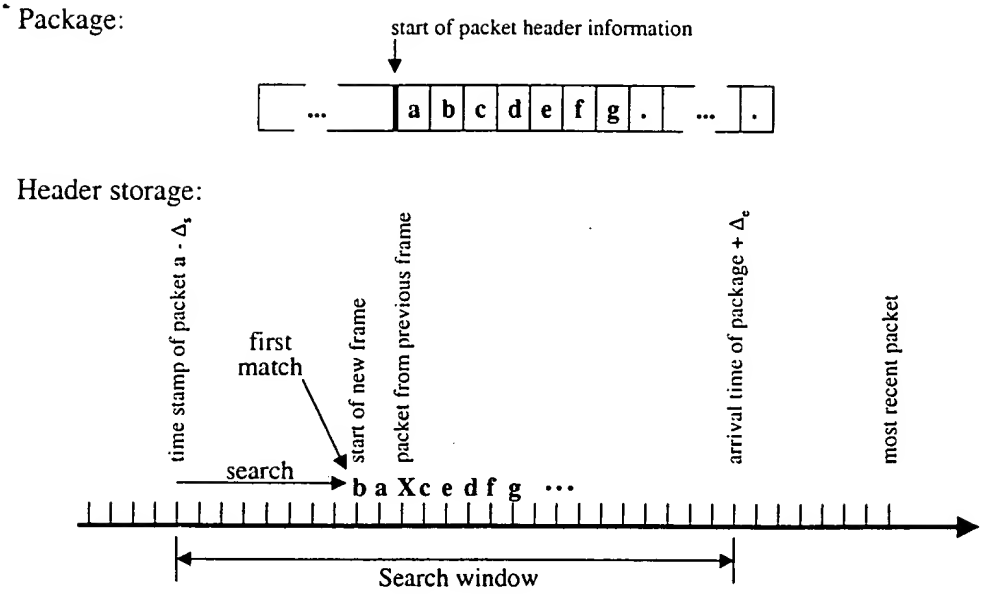

Figure 18: Searching for frame boundary when *out of frame*.

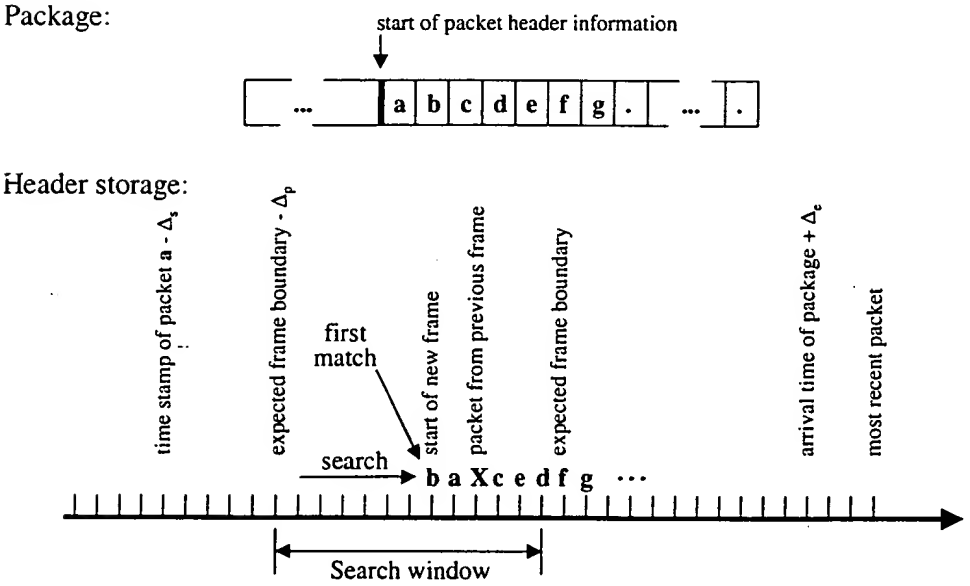

Figure 19: Searching for frame boundary when *in frame*.

Searching for headers in a given window (Lines 6, 25, and 40) is the computation-intensive portion of the algorithm. To find the first match for any of the package headers, we must step through the headers in the window comparing each to all of the package headers. It is desirable to have the smallest possible window and to use a small number of headers from the package for the search. If little loss is expected, we can use a subset of the headers for the search; as conditions in the network change (i.e., there is more or less loss) the number of headers used in the search can be increased or decreased. There are many ways the search could be optimized.

When the receiver is in frame and is unable to match several consecutive packages, it is changed to the out-of-frame state (lines 41-50 in Figure 17). The *badframes* variable counts the number of consecutive packages that could not be found in the header storage. When a package is not found, the size of the current frame is combined

with that of the next frame to compute the next expected frame boundary (line 18). The *frameoffset* variable stores the information needed to perform this computation (line 46).

Because the algorithm uses the first header in the storage that matches *any* of the packet headers in the package as the frame boundary, the algorithm implicitly deals with the case where packets arrive out of order. However, out-of-order packets could cause error in the packet loss measurements. Packets that were sent after the first package packet that matched the storage, but arrived after the first package packet will be counted as lost in one frame and as extra packets in the next frame (e.g., packet **X** in Figure 18). Packets sent after the first package packet that matched the storage but received before the first package packet (this cannot include later packets that are included in the package), will be counted as extra packets in one frame and lost in the next. Thus, there may be some error in the packet loss counts for individual frames, but the total packet loss when a sequence of frames is summed should be more accurate since all real losses will be counted and the miscounts will cancel out.

Key Ideas

The success of IPACA hinges on several crucial discoveries*: scalable architecture based on the sampled package idea; unique IP packet identification*; *implicit re-ordering of the receiving packet-trace*; and *duplicate-packet detection and processing*.

- The idea of sampling and creating a package at the sender is the key to reduce the overhead data capacity from 3% of the full rate to an insignificant fraction. This data reduction architecture is the key to large-scale deployment of SLA monitoring systems.

- IP packets can be uniquely identified using a tuple containing source address, source port, destination address, destination port, fragment flag and offset, and IP identifier. The IP identifier in every packet is a 16-bit field that is guaranteed to be unique within 64K packets, given the other members of the tuple above. This unique identification mechanism is the key to IPACA's ability to match IP headers between two packet-traces.

- IPACA synchronizes the frames between packet-traces using samples of packets taken at regular intervals. The reason we can use this approach is that we can theoretically rearrange the out-of-order output trace to produce the same order as the input trace. Instead of actually rearranging the output trace, which would require a large amount of time, space, bandwidth, and money, we can *assume* that we have rearranged the trace and synchronize the frames to a single match or a small set of matches. This allows us to consume far less resources and produces a solution that scales easily to large numbers of IP flows.
  There are two effects of this assumption: we cannot look into the future, to see if some out-of-order packets (that we have counted as missing) will be received before the IP timeout; and some packets sent before the first window may be received after it. Our measurements will tend to marginally under-count the frame-size at B for the first reason, and marginally over-count for the second reason. However, since these inaccuracies occur only for the first and last frames, this error vanishes into insignificance given any reasonable metric measurement time. Moreover, if the actual frame-size is reasonably large, the error can also be small for a single frame.

- We seek to eliminate the effects of duplicate packets in the framing algorithm. If a packet is duplicated within our measured ATM/FR/IP network, the duplicate should not be considered a part of the synchronized frame at B. However, if a packet is duplicated prior to entry (i.e., prior to A), we want to *include* the duplicate in both the synchronized frames. In this case our network is faithfully passing along the duplicated packets given to it. The former problem is solved by searching for duplicate packets in the B-trace, and not including these duplicates in the FS calculation. TO solve the latter problem we count packets that arrive already duplicated at A, and send this number as part of the package; then workstation B adds this number to the appropriate frame-size calculation to compensate for its over-counting of duplicates.

Some Applications of IPACA

IP Packet Loss

We can use IPACA to very accurately calculate IP packet loss over the measured network. Once we have synchronized the frames between MP A and MP B, we can calculate instantaneous PLR by dividing the frame-size of the B-frame by the frame-size of the corresponding A-frame. Calculation of an average PLR over a given time period is simply the total size of received frames in that period divided by the total size of sent frames in that period.

The implicit re-ordering error discussed above affects the PLR measurement, in particular the instantaneous measurement; however, this error can be diminished into insignificance by choosing an appropriately large frame-size. The duplicate-packet processing discussed above ensures that we are counting real packet loss in our network—unaffected by accidental packet duplications within our network or elsewhere.

IP Delay and Jitter

A slight extension of IPACA allows us to gather large samples of delay and jitter statistics from real traffic on the network. Instead of matching just one header from the package in B's header-storage, we match all the headers. Since the package contains time-stamps for each included header, and, since the headers in the package were sent sequentially, we can calculate delay for each packet and jitter for each pair of packets. The delay and jitter samples thus collected can be combined between frames to produce average statistics for both SLA metrics.

References

[Apisdorf96] Joel Apisdorf, k claffy, Kevin Thompson, and Rick Wilder, *OC3MON: flexible, affordable, high performance statistics collection*, September 13, 1996. Available at http://www.nlanr.net/NA/Oc3mon/

[Cisco99] Cisco Systems Inc., *Response Time Reporter Enhancements*, May 24, 1999. Available at http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120t/120t3/rtrenh.htm

[I.380] *ITU-T Recommendation I.380.*

[Krishnan96] K. R. Krishnan, *Auto-discovery in a frame relay network using MIBs under SNMP*, Project R6SS08.0F report, June, 1996.

[PerfPoint99] *Performance Point™ System Requirements: Requirements for Service Level Agreement Management Prototype*, Release 3.0, Issue 1, Telcordia Document BD-PMR-REQ-008, May, 1999.

[Surveyor97] *Surveyor Home Page*, 1997. Available at http://www.advanced.org/surveyor

[VisualNetworks99] *Visual Uptime*, April 14, 1999. Available at http://www.visualnetworks.com/product/prodmain.htm

[WAND99] WAND Research Group (Waikato University), *DAG3 Information*, May 6, 1999. Available at http://atm.cs.waikato.ac.nz/atm/docs/dag3/